Available online at www.sciencedirect.com

SciVerse ScienceDirect

Physics Procedia 25 (2012) 2006 – 2013

Physics Procedia

2012 International Conference on Solid State Devices and Materials Science

The Double-System Architecture for Trusted OS

Yong Zhao[1,2] , Yu Li[1,2], Jing Zhan[1]

[1]*College of Computer Science and Technology, Beijing University of Technology, Beijing 100124, China*
[2]*Key Lab of Information Network Security, Ministry of Public Security, Shanghai 201204, China*

Abstract

With the development of computer science and technology, current secure operating systems failed to respond to many new security challenges. Trusted operating system (TOS) is proposed to try to solve these problems. However, there are no mature, unified architectures for the TOS yet, since most of them cannot make clear of the relationship between security mechanism and the trusted mechanism. Therefore, this paper proposes a double-system architecture (DSA) for the TOS to solve the problem. The DSA is composed of the Trusted System (TS) and the Security System (SS). We constructed the TS by establishing a trusted environment and realized related SS. Furthermore, we proposed the Trusted Information Channel (TIC) to protect the information flow between TS and SS. In a word, the double-system architecture we proposed can provide reliable protection for the OS through the SS with the supports provided by the TS.

Keywords:Double System Architecture, Trusted Information Channel, Trusted Operating System, Trusted Computing

1. Introduction

Since the first Secure Operating System (SOS) Adept-50 [1] appeared, the SOS has been developing greatly in recent 40 years. SOS goes through foundation period, cookbook period, multi-policy period and dynamic-policy period [2]. With the rapid development of computer science and technology, SOS has been unable to respond to new security challenges. And some drawbacks of SOS have been gradually revealed.

Unable to ensure the credibility of subject. In SOS, the subject can be authorized to access the object legitimately. However, the restriction can only do access control, such as whether the subject can be access to the object, and what access the subject is authorized to. Whether the subject is trusted or whether the integrity of the subject is destroyed is difficult for SOS to solve.

Hard to determine the behavior of subject is trusted or not. When an access request coming, SOS can determine the subject's behavior does not obey the security policies and will not do harm to the security of the system. But SOS can not decide the behavior is expected as designed.

Unable to ensure the credibility of object. In the traditional SOS framework, it is easy to determine whether the subject access request meets the security policy. But the integrity of the object is not

1875-3892 doi:10.1016/j.phpro.2012.03.342

considered Take the dynamic link library (DLL) called by process for example. If the DLL is infected with virus, although the subject has no malicious purpose, the system security will be broken once the DLL is loaded.

In recent years, with the rapid development of Trusted Computing Group (TCG), trusted computing technology is able to verify the integrity of the subject and object, to measure the credibility of the subject behaviors, and sealed storage with Trusted Platform Module (TPM). Besides, the chain of trusted mechanism can ensure that the computer hardware and software are not replaced or broken in the boot process. Therefore, SOS is going to the direction towards the Trusted Operating System (TOS).

In the study of TOS, the key problem to be solved is the architecture. At present, there are no mature, unified architectures for TOS, because most of them can not make clear of the relationship between security mechanism and the trusted mechanism. Therefore, this paper proposes a double-system architecture (DSA) for TOS.

2. Related Work

2.1 Flask

The Flask architecture [3] cleanly separates the definition of the policy logic from the enforcement mechanism. The security policy logic is encapsulated within a separate component of the operating system with well-defined interfaces for obtaining security policy decisions. This separate component is referred to as the security server due to its origins as a user-space server running on a microkernel. Components in the system that enforce the security policy are referred to as object managers. Object managers are modified to obtain security policy decisions from the security server and to apply these decisions to label and control access to their objects.

2.2 Next Generation Secure Computing Base

In order to construct a new trusted computing platform, Next Generation Secure Computing Base (NGSCB) [4-6] is proposed by Microsoft. NGSCB has four characteristics. (i) Strong process isolation. The NGSCB application can be assured that it is not modified or observed by any other application or even the operating system. (ii) Sealed storage. Information can be stored in such a way that only the application from which data is saved (or a trusted designated application or entity) can open it. (iii) Secure path to and from the user. Secure channels allow data to move safely from the keyboard/mouse to NGSCB applications. (iv) Attestation. With attestation, a piece of code can digitally sign or otherwise attest to a piece of data and thus assure the recipient that the data was constructed by an unforgettable, cryptographically identified trusted software stack.

2.3 Terra

The key of Terra[7,8] is trusted virtual machine monitor (TVMM) which partitions a tamper-resistant hardware platform into multiple, isolated virtual machines (VM), providing the appearance of multiple boxes on a single, general-purpose platform.

Among the three architectures above, some are not supported by trusted computing, but they can not solve the three problems mentioned in Chapter 1. While others haven't merged the trusted computing mechanism and secure mechanism entirely, so trusted computing can not play the supporting role in system security. As a result, trusted mechanism and secure mechanism just do their own work each, and can not construct a trusted environment for the OS.

3. Double-system Architecture

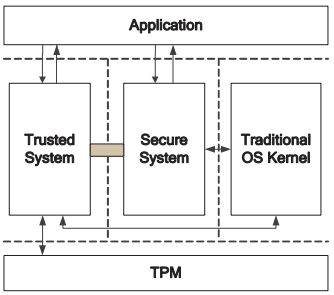

Figure 1. DSA

In order to solve the problems in Chapter 1, this paper proposes a new double-system architecture (DSA). Constructed based on the traditional OS kernel, the DSA is composed with secure system (SS) and trusted system (TS). The DSA is logically isolated with the traditional OS kernel, to prevent the DSA from being maliciously tampered. The TS is separated from the SS.

The TS provides trusted computing functions for the system through the TPM. It also provides the integrity verification function for the SS. The most important thing is it can prevent the system switching to an unexpected state through the trusted measurement. The SS authorizes the access request for the whole system and makes sure that the system runs in a security environment.

3.1 Trusted System

The TS is constructed with three modules: Trusted Measurement, Trusted Services and Trusted Report.

1) Trusted Measurement

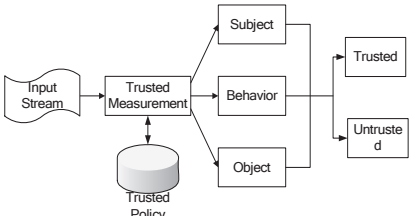

Figure 2. Workflow of Trusted Measurement

Trusted Measurement (TM) module does the measuring work on the subject, object and subject behavior (Fig 2). The input stream from the application layer, traditional OS kernel or SS is written as $(Sub, Obj, Behavior)$. Accordingly, the goal is to verify that whether the subject, object and behavior are trusted or not.

The TM does the measuring work on its own initiative when the process started. At this moment, the input stream is from application layer or the traditional OS kernel. However, for the input stream from the SS, the TM will be called by the SS to verify the integrity of the entity in $(Sub, Obj, Behavior)$ passively.

The technology of TM will use the popular methods, such as binary measurement [9,10] and behavior-based measurement [11]. Trusted Policy is the knowledge library, including the information about subject and object, the checksum of the programs, the relationship between the modules of the applications. The result of TM is written as $\langle Trusted, Untrusted \rangle$.

It is extremely important to prevent the information flow from being maliciously tampered. Take the following situation for example.

The pre-defined security policies are as follows.

if $Tom \rightarrow trusted$ *,then* $(Tom, a.txt, write) \rightarrow Permit$ *if* $Tom \rightarrow Untrusted$ *,then* $(Tom, a.txt, write) \rightarrow Forbid$

If the normal result Tom is not trusted is maliciously modified to Tom is trusted, the result of trusted measurement will get a fundamental change. In order to solve this problem, a new Trusted Information Channel (TIC) is proposed (in Chapter 3.3).

2) Trusted Services

Trusted Services module provided interfaces for encryption and decryption by TPM.

3) Trusted Report

Trusted Report module sends the trusted measurement results or PCR value to SS.

3.2 Secure System

The SS is constructed with four modules: User Identification, Access Control, Isolation Mechanism and Audit.

4) User Identification

The User Identification module checks the validity of the user's identity. Compared with the local login procedure, the check for network user must get the TPM information and PCR value from the remote terminal to make decision.

5) Access Control

The Access Control module is constructed with three modules: Hook, Security Policy Library and Decision Mechanism.

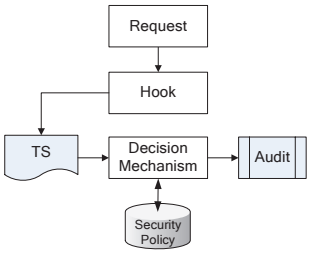

Figure 3. Workflow of the SS

a) Hook module hooks the access request.

b) Hook module sends the information to TS.

c) TS measures the subject, object and behaviors. Then, it sends the result to the Decision Mechanism module.

d) If any item in the result is not trusted, Decision Mechanism refused the request. Skip to f.

e) Decision Mechanism queries Security Policy Library. Then, determine whether the access request is permitted or not.

6) Isolation Mechanism

The Isolation Mechanism has three major functions. The first, it separates the important applications and prevent different applications from interfering to ensure the confidentiality of the system is not broken. The second, to prevent unauthorized access or malicious tampered, it provides the logical isolation for the DSA from the traditional OS kernel. The third, this approach will not only logically separate from the TS and SS, so that the relationship between secure mechanism and trusted mechanism is clearer, and can effectively prevent illegal access to TS.

7) Audit

The Audit module records the access request and the result in detail.

3.3 DSA Features

8) Relationship between TS and SS

Through the trusted measurement, TS makes sure of the reality of subject, object and the subject behavior. Besides, TS provides a trusted environment for the implementation of security policy in SS to prevent the system entering an unexpected state. Without such a trusted environment, the decision mechanism in SS will be meaningless. If a malicious process imitate a trusted process and the trust process is authorized to access some special important resources, the access control mechanism can not guarantee the system security is not broken. If the application which is infected with virus does self-copy ceaselessly, the process behavior does not obey the security policy, but it is unexpected. In these cases, for the SS can not resist the attack alone, the key effect of the TS is revealed. The TS sustains SS effectively.

The SS provides the secure mechanism for access control, logical isolation, etc. By prohibiting the unauthorized access, a secure environment is constructed. With the trusted environment provided by TS, the SS confirms the security of OS. With the isolation mechanism, the SS can prevent the TS from being maliciously tampered. Otherwise, the unauthorized request may result in the bypass or broke of the TS. The SS ensures that the SS can work normally.

9) The information protection based on Trusted Information Channel

TS is logically isolated with SS. The transmission of information between TS and SS is "Default Deny". The communication between the two must be completed with Trusted Information Channel (TIC). The structure of TIC is depicted as below (Fig 4).

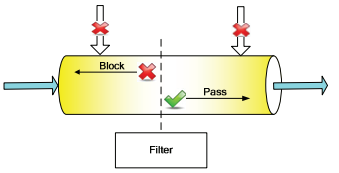

Figure 4. Structure of TIC

TIC has two functions:

a) Trusted transmission. After TIC is established with the isolation mechanism, any external access to TIC will be prohibited. Besides, encryption is used in TIC. The information entering the TIC is encrypted, and decrypted when it is leaving the TIC. The checksum value is attached to the trail of packet.

b) Block the unexpected information flow. By Filter module, only several pre-defined types of information flow can pass through TIC, others will be blocked.

With TIC, Only the expected information flow can be sent form SS to TS. With encryption, the information is hard to hook. And it is difficult to modify the information by using the checksum.

4. DSA Implementation

In our experiment, a file system filter driver and a device filter driver in Windows XP are developed based on the DSA. The analysis on DSA functionality and performance are also given.

4.1 SS Construction

We developed a file system filter driver according to the DSA. Firstly, the request of process loaded to the memory is obtained through the callback PsSetLoadImageNotifyRoutine. Using the structure

$(\mathrm{Pr}\,ocessID, \mathrm{Pr}\,ocessFullPath)$, add the record to a LIST_ENTRY. Secondly, get the access request by dispatch IRP_MJ_CREATE. In the dispatch function, call PsGetCurrentProcessId to get the subject's ProcessID, check LIST_ENTRY to get the full path name of the process. Thirdly, using FileObject in I/O Request Packet (IRP), object can be retrieved. At last, through DesiredAccess in SecurityContext, get the operation type. So, we realize the hook module.

The Security Policy Library is recorded in file SecurityPolicy.skr. Decision Mechanism is the return value of the dispatch function. Audit module writes the records in file Audit.skr.

4.2 TS Construction

In the file system filter driver, White-List technology is used to verify the integrity of the executable codes (such as, .Exe, .Dll, .Com, etc.). We use ThinkPad X61t Atmel TPM. At Initial state, With the structure $(Fullpath, Hash)$, all checksum values of executable codes are collected in file FileInfoList.skr.

When TS receives $(Sub, Obj, Behavior)$ from SS, it will check the integrity of subject with checksum value. If the object is an executable program, TS will verify the integrity of object. Finally, Trusted Report module will send the results to SS. In this way, the reality of subject and object will be checked.

Trusted Services module call the TPM interfaces directly, including encryption, decryption, hash calculation, and so on.

4.3 TIC Construction

We also developed a device filter driver in the experiment. By calling IoAttachDeviceToDeviceStack, the device filter driver mounted to the upper level of TPM driver. Through the dispatch function IRP_MJ_DEVICE_CONTROL and IRP_MJ_INTERNAL_DEVICE_CONTROL, we hook the IRP which accesses the TPM driver.

The information which from SS is encrypted and is added integrity check information. Correspondingly, device filter driver will decrypt the received data and verify the integrity. In this way, the TIC is constructed.

4.4 Security Analysis

With the DSA, the credibility of subject and object can be ensured, and whether the behavior is trusted or not can be determined. The indispensable two systems in the DSA cooperate with each other, and they can solve the three problems mentioned in Chapter 1 well. Running on the trusted environment, the two systems do the work for OS security and integrity together.

4.5 Performance

The influences of DSA are mainly on the integrity verification and encryption of transmission information. Therefore, we construct a testing program Q as follows: First, choose a program o for test. Second, create three threads which complete the operation ($Start\ \ o$, $Kill\ \ o$) 100 times. At last, get the time which Q costs.

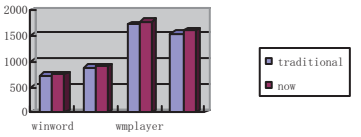

Figure 5. Efficiency Influence

The efficiency of the OS decreased only 3.2 percent, which is acceptable on user experience.

5. Conclusion

We construct the Trusted System by establishing a trusted environment. The Secure System is also realized with the security mechanism we proposed. Furthermore, we propose the TIC to achieve protection for the information transmission between TS and SS. In a word, the Double-System Architecture can provide the reliable protection for the OS through the SS with the supports provided by the TS. Besides the secure mechanism, the DSA can check the integrity of the subject, object and the behavior. And it makes clear of the relationship between security mechanism and the trusted mechanism.

6. Acknowledgment

This work is partially supported by 863 National High Tech Research and Develop Plan Project #2009AA01Z437, 973 National Key Fundamental Research Development Plan Project #2007CB311100 and Key Lab of Information Network Security, Ministry of Public Security.

References

[1] Clark Weissman. Security Controls in the ADEPT-50 Time Sharing System. Proceedings of the 1069 AFIPS Fall Joint Computer Conference, AFIPS Press, 1969, pp.119-133.

[2] Wenchang Shi. The development of Research on Secure Operating Systems. Computer Science, 2001.

[3] Secure Computing Corporation. Assurance in the Fluke Microkernel: Final Report. CDRL Sequence NO.A002, Secure Computing Corporation, Apr 1999.

[4] A.Carroll , M.Juarez , et al. Microsoft Palladium: A business overview. http://www.microsoftcom/PressPass/features/2002/juI02/0724Palladiumwp.asp, Aug 2002.

[5] Microsoft , Microsoft Next Generation Secure Computing Base: An Overview , http://www.microsoft.com/resource/ngscb/ngscb_overview.mspx, Apr 2003.

[6] P.England, B.LamPson, et al. A Trusted Open Platform. IEEE Compute, 36(7):55-62, 2003.

[7] T.Gafinkel, M.Rosenblum, D. Boneh. Flexible 0S support and applications for trusted computing. HotOS-IX, 2003.

[8] T.Garfinkel, B.Pfaff, et al. Terra: a virtual machine-based platform for trusted computing. Proeeedings of the nineteenth ACM symposium on Operating systems principles, ACM Press, 193-206, 2003.

[9] Fred Cohen. Computer Viruses-Theory and Experiments. Computers and Security. 1984

[10] Fred Cohen. Computer Viruses. PhD thesis, University of Southern California, 1985

[11] Nick L. Petroni, Jn. Timothy Fraser. Copilot - a Coprocessor-based Kernel Runtime Integrity Monitor. Proceedings of the13th USENIX Security Symposium. August 9–13, 2004

[12] Chen P M, Noble B D. When Virtual is Better Than Real[C]. 8th Workshop on Hot Topics in Operating Systems (HotOS'01), 2001: 133-138.

[13] Matt Bishop. Computer Security: Art and Science. Addison Wesley Professional. 2003:406~411

[14] Tang Ziying. Computer Operating System. Press of University of Electronic Science and Technology of Xi'an, China, 2001-08

[15] Whitaker A, Shaw M, Gribble S D. Denali: A Scalable Isolation Kernel[C]. Proceedings of the 10th ACM SIGOPS European Workshop, 2002: 10-15